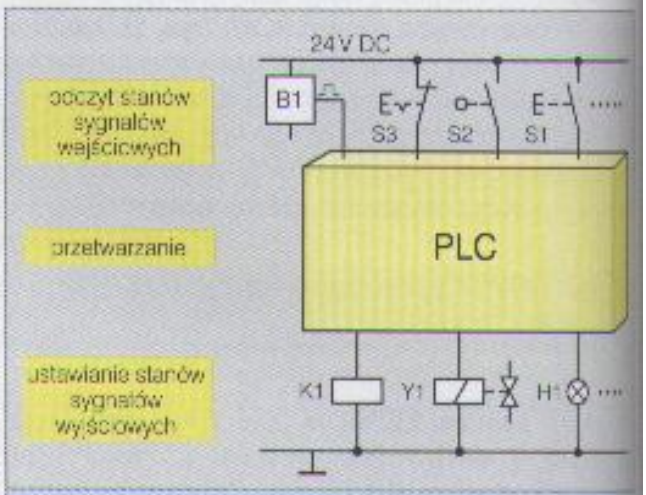
2.6 Sterowniki programowalne (PLC)

2.6.1 Budowa i zasada działania

Sterownik programowalny **PLC**[1] jest zbudowany podobnie jak komputer. Składa się z zasilacza (układu zasilania napięciowego), modułu sygnałów wejściowych, jednostki centralnej z mikroprocesorem, pamięci programu oraz modułu sygnałów wyjściowych. Inaczej niż w układach stałoprogramowych, w których przebieg sterowania jest określony przez wykorzystane elementy i ich wzajemne połączenia, w sterowniku programowalnym przebieg ten jest zapamiętany w pamięci sterownika w postaci programu.

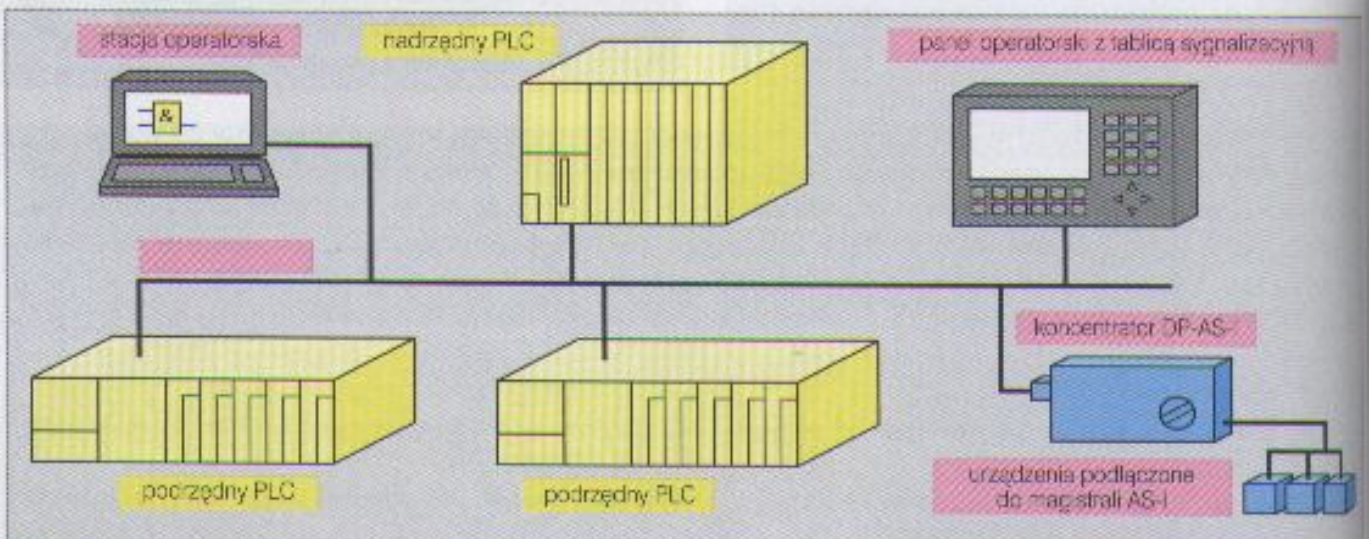
Sterownik programowalny jest wykorzystywany jako część centralna, przetwarzająca, układu sterowania (**rys. 1**). W układach o niewielkiej głębokości przetwarzania, odpowiadającej maksymalnie 100 **DI/DO** (DI, ang. ***D**iscrete **I**nput* = wejście dyskretne, dwustanowe; DO, ang. ***D**iscrete **O**utput* = wyjście dyskretne, dwustanowe) wykorzystuje się tzw. małe, kompaktowe sterowniki, nazywane także mikrosterownikami (**rys. 2**); w przypadku złożonych zadań sterowania – sterowniki nazywane średnimi (100 – 500 DI/DO) lub dużymi (500 – 3000 DI/DO), o budowie modułowej.

Dla obsługi poszczególnych jednostek produkcyjnych wykorzystuje się często sterowniki lokalne, „podrzędne" (*Slave*), połączone przy pomocy sieci ze sterownikiem „nadrzędnym" (*Master*[2]), np. PROFIBUS-DP. Do magistrali (ang. *BUS*[3]) może być podłączonych wiele urządzeń (elementów sieci), jak np. sterowniki, stacje operatorskie i panele operatorskie. Mogą być także podłączone inne sieci, np. AS-I[4], pozwalające na bezpośrednią współpracę z urządzeniami pomiarowymi i wykonawczymi w systemie rozproszonego przetwarzania danych (**rys. 3**).



Rys. 1. Struktura układu sterowania programowalnego

Rys. 2. Mikrosterownik PLC



Rys.3. Sieć sterownikowa

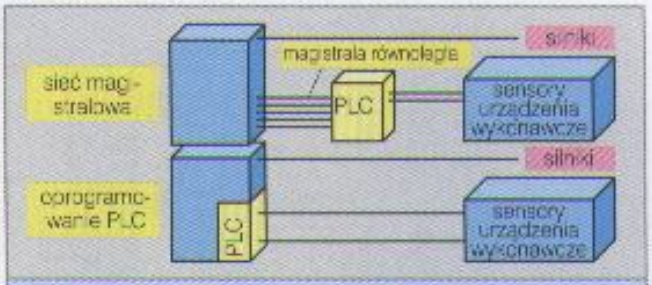
[1] **PLC**, ang. *Programmable Logic Controller*; [2] ang. *Master-Slave* = pan-sługa; [3] ang. *Bus* = zbiorcza szyna, magistrala dla wymiany danych
[4] **AS-I**, ang. *Actuator-Sensor-Interface*

Sterownik programowalny może być także zintegrowany z innym układem sterowania cyfrowego (mikroprocesorowego), np. z komputerowym sterownikiem numerycznym **CNC** (ang. ***C**omputer **N**umerical **C**ontrol*). W tym przypadku sterownik programowalny będzie aktywniany przez program sterownika CNC (**rys. 1**). Jeśli potrzebne są tylko wybrane funkcje sterownika programowalnego, to najczęściej integruje się tylko odpowiednie moduły oprogramowania z systemem operacyjnym sterownika CNC. Coraz częściej też, wraz z wykorzystaniem komputerów typu PC[1] do programowania sterowników i do sporządzania dokumentacji programu oraz z wprowadzaniem rozwiązań sprzętowych i programowych stosowanych w komputerach do systemów sterowania, wykorzystuje się komputery przemysłowe dla realizacji zadań sterownika programowalnego (**rys. 2**). W tym przypadku możliwe jest bądź wykorzystanie komputera przemysłowego z oprogramowaniem PLC bądź wykorzystanie w komputerze kart rozszerzeń typu PLC.

Sterowniki programowalne PLC spotyka się w postaci kompaktowej, modułowej, połączonej za pomocą sieci, zintegrowanej z innymi sterownikami i w postaci komputera przemysłowego.

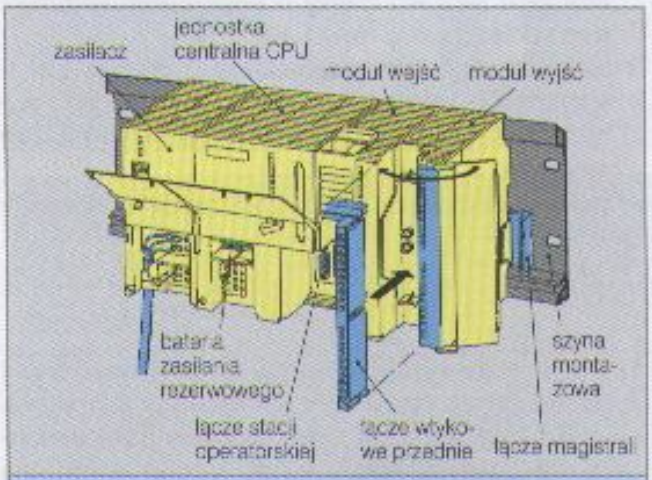
Obecnie najczęściej wykorzystuje się sterowniki PLC o budowie modułowej. **Centralną grupę modułów** sterownika tworzą zasilacz, moduł jednostki centralnej CPU[2] oraz przynajmniej jeden moduł wejść dyskretnych i jeden moduł wyjść dyskretnych (**rys. 3**).
Moduł jednostki centralnej CPU należy podłączyć do źródła napięcia, przeważnie jest to 24 V DC (**rys. 4**). Aby nie powodować utraty zapamiętanych danych podczas awarii zasilania, podtrzymuje się zasilanie modułu za pomocą baterii lub akumulatorów.
Jeśli przełącznik rodzaju pracy jest w pozycji STOP, programy aplikacyjne mogą zostać przesłane do jednostki CPU. Wymianę danych pomiędzy stacją operatorską i modułem CPU zapewnia zwykle interfejs komunikacyjny typu MPI[3]. Pozycji MRES[4] przełącznika modułu odpowiada kasowanie pamięci CPU; pozycji RUN – przetwarzanie programu aplikacyjnego; pozycji RUN-P – przetwarzanie z możliwością ustawienia wartości zmiennych.
Przez interfejs magistrali PROFIBUS-DP moduł CPU komunikuje się z innymi modułami – moduły te nazywane są peryferyjnymi.
Wskaźniki systemowe informują o aktualnym stanie pracy modułu CPU. Przy wskazaniu SF lub SF DP dokładne informacje o rodzaju popełnionego błędu mogą być pobrane z modułu kontrolera. Jednostka centralna zawiera **mikroprocesor** oraz różne obszary pamięci – obszar **danych, roboczy i systemowy**, połączone z sobą wewnętrzną magistralą sterownika.

[1] **PC**, ang. *Personal Computer* = komputer osobisty; [2] **CPU**, ang. *Central Processing Unit* = centralna jednostka przetwarzająca; [3] **MPI**, ang. *Multi Point Interface*; [4] **MRES**, ang. *Memory Reset*

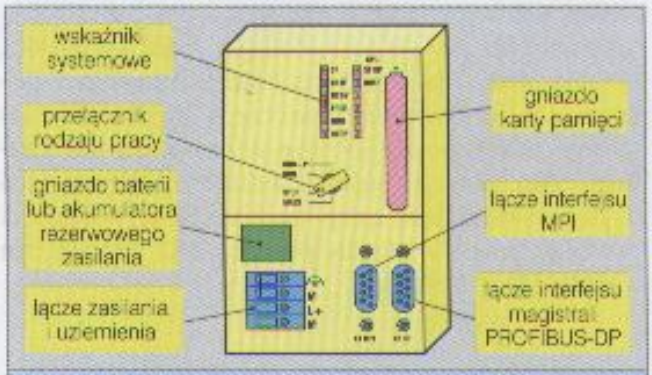


Rys. 1. Zintegrowane sterowniki programowalne

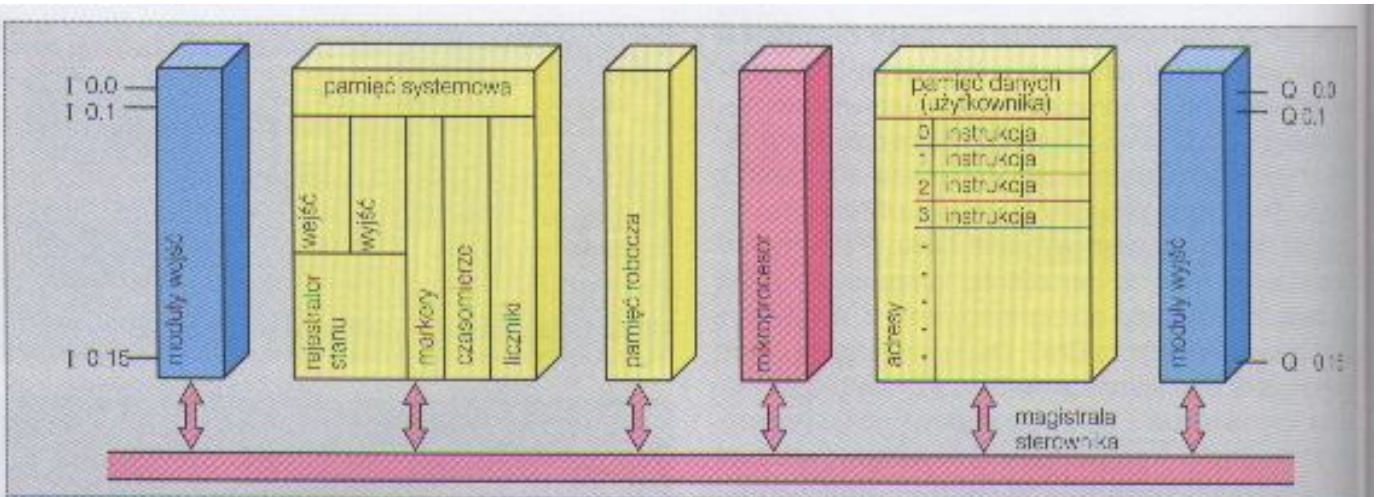

Rys. 2. Komputer przemysłowy



Rys. 3. Sterownik programowalny o budowie modułowej



Rys. 4. Płyta czołowa jednostki centralnej CPU



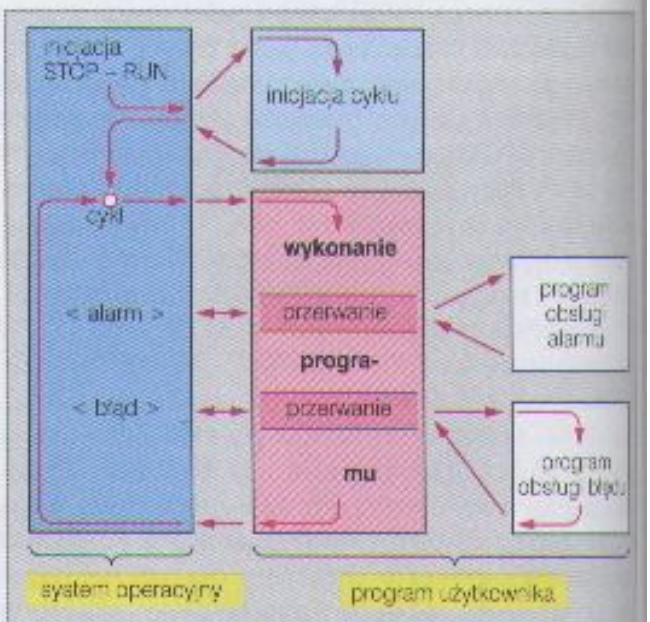
Rys. 1. Budowa wewnętrzna sterownika PLC

W **pamięci danych** sterownika przechowywane są dane i instrukcje programu użytkownika (**rys. 1**). Pamięć ta jest albo typu RAM[1], albo EPROM[2] – może być rozszerzona za pomocą dodatkowych kart lub modułów pamięci. W przypadku karty wyposażonej w pamięć EPROM, program użytkownika może być do niej załadowany w trybie off-line[4], co zabezpiecza go przed utratą podczas awarii zasilania. **Pamięć robocza** jest szybką pamięcią typu RAM. Kopiowane są do niej dane w trakcie przetwarzania programu użytkownika. Pamięć systemowa zawiera zmienne, nazywane także operandami, na których wykonywane są operacje programu. Zmienne te zbierane są w wydzielone obszary, nazywane obszarami operandów. Wielkość tych obszarów zależy od zastosowanego CPU.

Operandy CPU:

- **zmienne wejściowe, wejścia** (**I**, *ang. Input*) – dostarczane jako argumenty przetwarzania programowego przez moduły wejściowe,
- **zmienne wyjściowe, wyjścia** (**Q**, z ang. *to quit* = opuszczać) – argumenty użyte w celu zwrócenia wyniku przetwarzania programowego przez moduły wyjściowe,
- **zmienne markujące, markery** (**M**, *ang. Marker – człowiek notujący zapisy w grze*) – zmienne wewnętrzne informujące o stanie przetwarzania,
- **zmienne czasowe** (**T**, *ang. Timer*) – generowane przez bloki funkcyjne wykorzystywane do realizacji uwarunkowań czasowych lub odmierzania czasu,
- **zmienne licznikowe** (**C**, *ang. Counter*) – przetwarzane przez bloki funkcyjne realizujące programowe liczenie dodające i odejmujące.

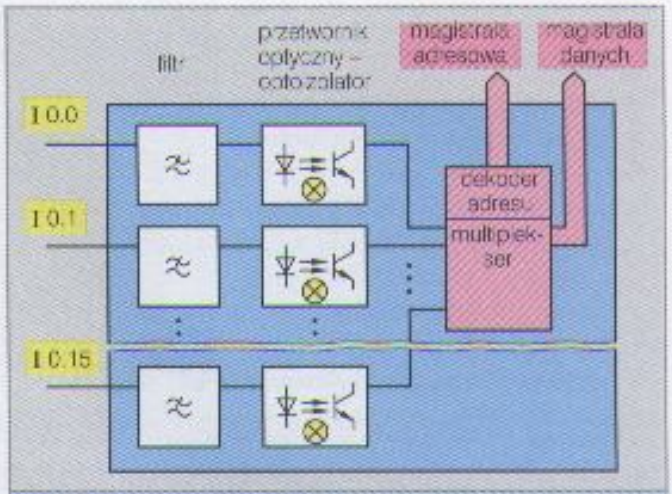
W stanie włączenia (RUN) program użytkownika jest cyklicznie przetwarzany (**rys. 2**). Najpierw odczytywany jest z pamięci systemowej aktualny **stan zmiennych wejściowych**. Potem następuje, odpowiednio do programu użytkownika, właściwe przetwarzanie. Wynikiem przetwarzania programowego jest nowy **stan zmiennych wyjściowych** zapamiętany w pamięci systemowej i podany do odpowiednich wyjść sterownika. Kolejno następuje aktualizacja stanu zmiennych wejściowych, przetworzenie, zwrócenie wyniku itd. Czas takiego pojedynczego przebiegu nazywany jest **czasem cyklu**. Jest tym większy, im dłuższy jest program użytkownika. Zależy oczywiście także od prędkości przetwarzania wykorzystywanego CPU. Przeważnie wynosi kilka milisekund.



Rys. 2. Przetwarzanie programu użytkownika

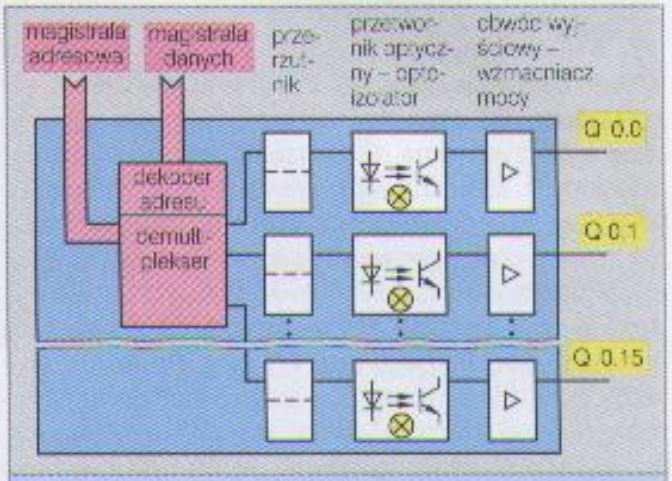
[1] **RAM** ang. *Random Access Memory* = pamięć ze swobodnym dostępem; [2] **EPROM** ang. *Erasable Programmable Read Only Memory* = elektrycznie programowalna pamięć typu ROM; off line, ang., tutaj w znaczeniu na zewnątrz CPU

Część wejściowa sterownika podzielona jest na **moduły** obejmujące przeważnie 8, 16 lub 32 wejść binarnych (**rys. 1**). Moduł wejściowy zawiera układy elektroniczne zamieniające sygnały pochodzące z urządzeń zewnętrznych na sygnały logiczne akceptowane przez sterownik. Mogą to być np. dzielniki napięć z dodatkowymi filtrami RC dla tłumienia zakłóceń. Moduły wejść prądu stałego wyposażone są dodatkowo w diody chroniące właściwą polaryzację (najczęściej dodatnią, nazywaną także „ze wspólnym plusem"), a moduły wejść prądu przemiennego – w mostkowe układy prostownicze. Dla izolacji potencjałowej obwodów wejściowych i magistrali sterownika stosowane są połączenia optyczne – optoizolacje (fotodioda jako nadajnik i fototranzystor jako odbiornik w jednym elemencie, tzw. optoizolator). Stan poszczególnych bitów bufora danych modułu wejściowego sygnalizowany jest diodami typu LED – pozwala to na szybką identyfikację stanu wejść w trakcie uruchomienia sterownika lub w trakcie poszukiwania błędów. Multiplekser (rozdzielacz) sterowany jest przez dekoder adresów.



Rys. 1. Schemat podłączeń wejść

Część wyjściowa sterownika podzielona jest również na moduły obejmujące 8 lub 16 wyjść binarnych (**rys. 2**). Moduł wyjściowy zawiera układy wzmacniające, np. łącznik tranzystorowy dla obwodów wyjściowych prądu stałego (np. 24 V DC, 200 mA) lub łącznik triakowy (elektroniczny łącznik prądu przemiennego) dla bezpośredniego wysterowania obwodów wyjściowych z obciążeniami prądowymi, indukcyjnymi i pojemnościowymi (np. 50 Hz, 220 V AC).



Rys. 2. Schemat podłączeń wyjść

W przypadku zakłócenia przetwarzania programu użytkownika, moduły wyjściowe po zakończeniu danego cyklu przetwarzania przerywają sterowanie dołączonymi obwodami wyjściowymi z urządzeniami wykonawczymi procesu – zapobiega to niebezpiecznym skutkom awarii.

Stan poszczególnych wyjść modułu określa demultiplekser sterowany przez CPU sterownika. W dekoderze adresów zostaje odkodowany adres wybranego przez mikroprocesor wyjścia i odpowiednia wartość binarna przesłana z magistrali danych przez demultiplekser do układów wyjściowych modułu.

2.6.2 Programowanie

2.6.2.1 Języki programowania

W związku z oczywistą koniecznością standaryzacji sterowników PLC w 1993 r. pod auspicjami Międzynarodowej Komisji Elektroniki (**IEC**, ang. *International Elektronical Commision*) została opracowana i wydana norma **IEC 1131** (**tabl. 1**). Norma podaje informacje ogólne i zalecenia w zakresie sprzętu, wymagań testowych, języków programowania, określa wytyczne dla użytkowników oraz zalecenia w zakresie wymiany danych. Nie ma charakteru wiążącego.

Tabl. 1. Norma IEC 1131 „Sterowniki Programowalne"

część nr	nazwa	treść
część 1	informacje ogólne	– definicje pojęć – cechy funkcjonalne PLC
część 2	sprzęt i wymagania testowe	– wymagania elektryczne – wymagania mechaniczne – wymagania funkcjonalne – badania i testy – warunki użytkowania – reklamacja sprzętu
część 3	języki programowania	– model programowy – model komunikacyjny – język programowania tekstowe – graficzne
część 4	wytyczne użytkownika	– analiza systemowa – wybór sprzętu – eksploatacja i konserwacja
część 5	wymiana danych	– komunikacja pomiędzy sterownikami różnych producentów – komunikacja pomiędzy sterownikami i innymi urządzeniami – komunikacja sieciowa – wymiana danych – stany alarmowe – administracja sieci

Wyróżnia się dwie podstawowe grupy języków programowania:

- grupę **języków tekstowych** oraz
- grupę **języków graficznych**.

Do grupy języków tekstowych zalicza się:

- **języki list instrukcji** (**IL**, ang. *Instruction List*),
- **języki strukturalne** (**ST**, ang. *Structured Text*),

do grupy języków graficznych:

- **języki schematów drabinkowych** (**LD**, ang. *Ladder Diagram*),
- **języki schematów blokowych** (**FBD**, ang. *Function Block Diagram*).

Oprócz tych grup języków stosowane są powszechnie w technice sterowników programowalnych dwie metody modelowania i programowania sekwencyjnych procesów produkcyjnych: **metoda Grafcet** podana w normie IEC 848 oraz **metoda SFC** (ang. *Sequential Function Chart*), nazywana także metodą **grafów sekwencji**, objęta normą IEC 1131–3. Obie metody stanowią narzędzie wykorzystywane do tworzenia programów w wymienionych, standardowych językach programowania – właściwie są zorientowanymi graficznie metodami zapisu algorytmu procesu wymagającymi określenia logicznych zależności przyczynowo–skutkowych przebiegu sterowanego procesu. Wspólną cechą obu metod jest formalizm **sieci Petriego typu P/T** (ang. *Position/ Transition*), przy czym „młodsza" z metod, metoda SFC, została w dużej mierze oparta o „starszą", o kilkanaście lat, metodę Grafcet – odnosząc się do opisu procesu i sterowania nie ma pomiędzy nimi większych różnic. Te podane podziały, nazwy języków i metod opisu nie są jednak aprobowane przez wszystkich producentów.

Do najszerzej stosowanych języków należą języki drabinkowe (LD), języki list instrukcji (IL) oraz języki schematów blokowych (FDB). Stosowane są w nich powszechnie znane, proste instrukcje i symbole, wykorzystywane także w innych technikach sterowania oraz w językach programowania komputerów.

Programowanie w języku schematów drabinkowych jest bardzo podobne do tworzenia schematów stykowo–przekaźnikowych układów sterowania elektrycznego.

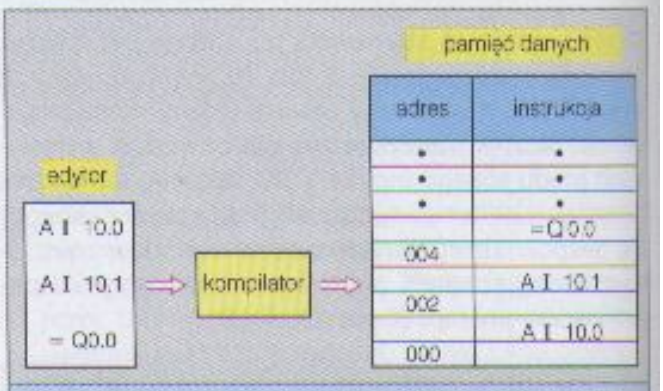
Programowanie w oparciu o metodę Grafcet i grafy sekwencji (SFC) wychodzi z opisu zadań sterowania sekwencyjnego za pomocą grafów zawierających etapy (kroki) i warunki przejścia (tranzycje) między tymi etapami.

Języki list instrukcji są najbardziej uniwersalną grupą języków programowania sterowników PLC.

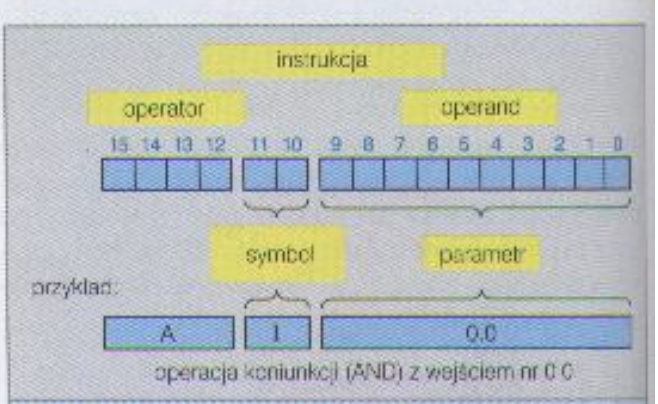
W trakcie wprowadzania programu użytkownika do sterownika wszystkie instrukcje muszą zostać przetłumaczone przez kompilator (ang. *to compile* = kompilować, tłumaczyć) na kod maszynowy zrozumiały dla sterownika PLC. Instrukcja zawiera nazwę operatora z ewentualnymi modyfikatorami oraz operand – argu-

Tabl. 1. Przykłady operacji wykonywanych przez sterowniki PLC wg IEC 1131–3

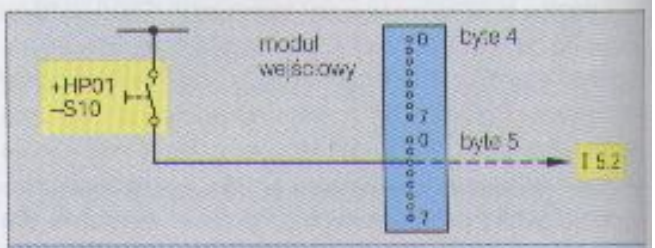
nazwa	zapis		
	IL	FBD	LD
I (AND)	U/A	&	
LUB (OR)	O	≥1	
NIE (NOT) (wejście)	N		
NIE (NOT) (wyjście)	N		
przypisanie	=	=	
ustawianie	S	S	S
zerowanie	R	R	R



Rys. 1. Wprowadzanie programu użytkownika do sterownika



Rys. 2. Format instrukcji



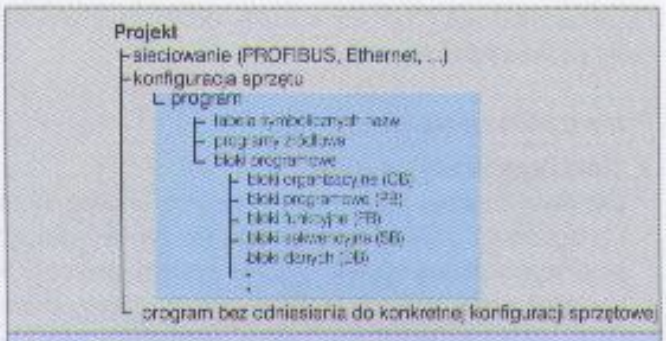
Rys. 3. Parametry operanda wejściowego

ment (**rys. 2 na poprzedniej stronie**). **Operator** określa działania, które mają być wykonane, **argument** – stałe lub zmienne poddane temu działaniu, składające się z **symbolu** i **parametru**. Symbol określa **typ zmiennej**, parametr podaje **adres zmiennej** (**rys. 3 na poprzedniej stronie**).

2.6.2.2 Budowa programu

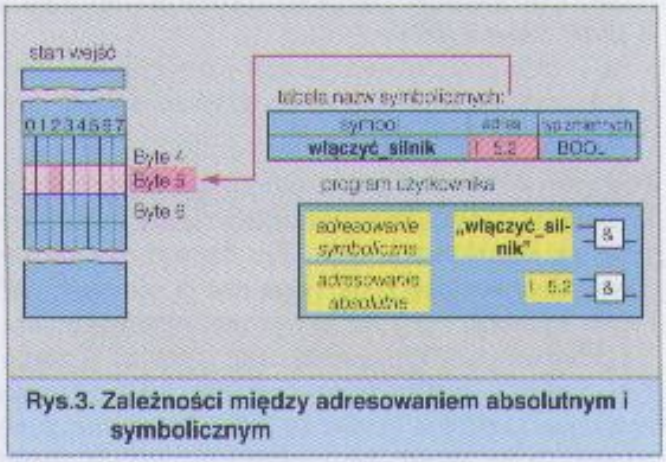
Zgodnie z normą IEC 1131 każde zadanie związane z automatyzacją traktowane jest jako *projekt*. Realizacja projektu przebiega zgodnie z określoną wstępnie *hierarchią zadań* do wykonania (**rys. 1**). Ponieważ oprogramowanie PLC jest zorientowane obiektowo, poszczególne części tworzonego w projekcie programu są obiektami, którym użytkownik przypisuje określone właściwości. Ogólnie realizacja projektu rozpoczyna się od ustalenia powiązań sieciowych. Kolejnym zadaniem jest wybór sprzętu, jego kompletacja i konfiguracja (**rys. 2**) – włączając zasilacz, moduł jednostki centralnej oraz moduły dyskretnych wejść i wyjść składające się na sterownik PLC. Skonfigurowanie sprzętu zamyka etap tworzenia struktury sprzętowej układu sterowania. Kolejnym etapem jest wybranie struktury programu użytkownika – jest on podzielony na oddzielne bloki programowe. Możliwe jest tu wykorzystanie istniejących programów źródłowych i typowych bloków programowych – także tych, które zostały napisane wcześniej, bez odniesienia się do konkretnej konfiguracji sprzętowej.

Korzystne dla użytkownika jest **adresowanie symboliczne** zmiennych (**rys. 3**) – łatwiej rozpoznać konkretne nazwy niż podobne do siebie ciągi cyfr **adresowania absolutnego**, np. I 124.5 oraz I 125.4. Po zadeklarowaniu zmiennej jako **zmiennej globalnej** jej symboliczna nazwa (**rys. 4**) jest znana we wszystkich blokach programu. Zmienna zadeklarowana w danym bloku i jej symboliczna nazwa jest – przeciwnie niż poprzednio wymieniona – znana tylko lokalnie (**zmienna lokalna**). Odpowiednie deklaracje używanych zmiennych powinien zawierać każdy blok programowy.

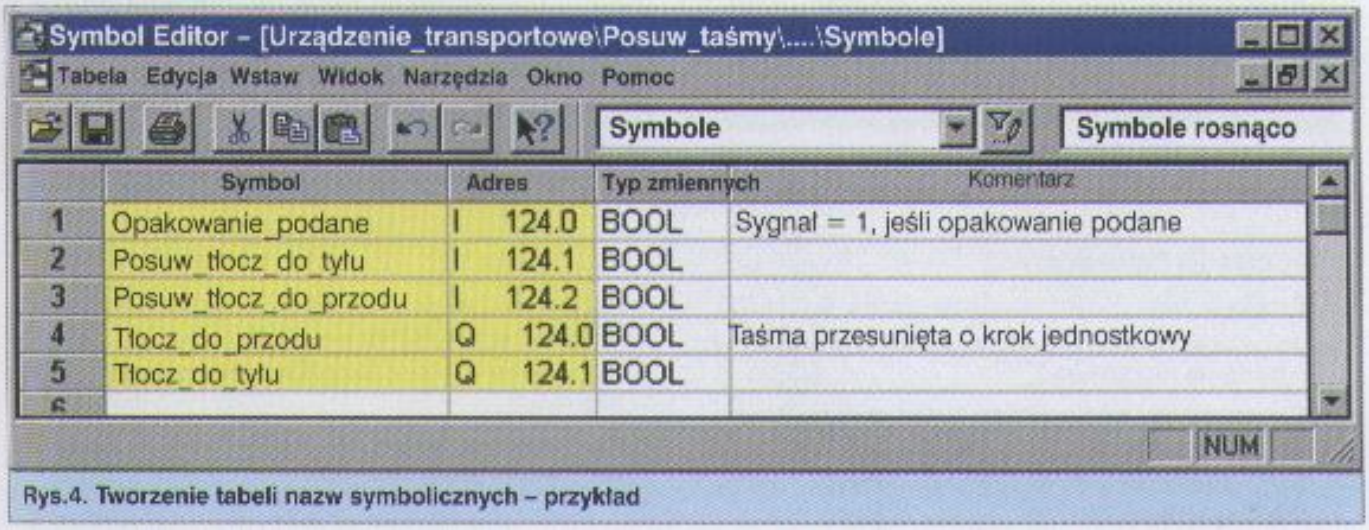


Rys. 1. Hierarchia zadań projektu

Rys. 2. Przykład konfigurowania sprzętu



Rys.3. Zależności między adresowaniem absolutnym i symbolicznym



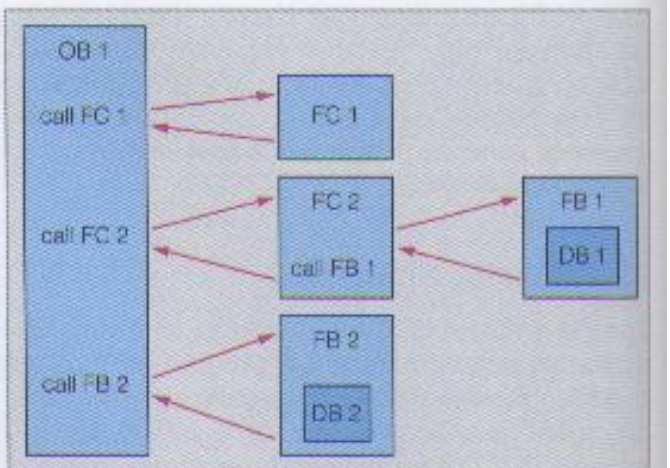
Rys.4. Tworzenie tabeli nazw symbolicznych – przykład

Użytkownik dzieli program w małe, dobrze przejrzyste bloki struktury modularnej (**rys. 1**).

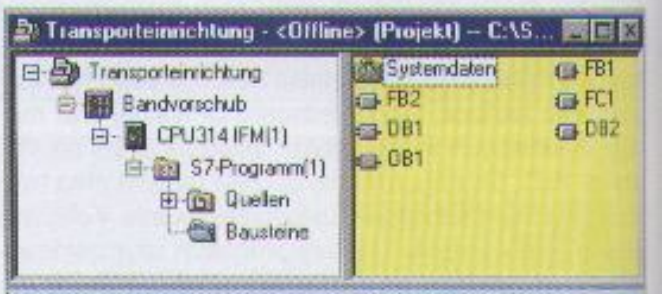
Przewidziane są następujące bloki:

- **bloki organizacyjne (OB)**
  stanowią interfejs pomiędzy systemem operacyjnym i programem użytkownika. Dzielą się na trzy grupy: bloki wywoływane cykliczne przez system operacyjny – w bloku takim znajduje się program główny (blok OB1), bloki wywoływane przez system operacyjny po wystąpieniu określonych zdarzeń (np. przerwania), bloki wywoływane w programie użytkownika,
- **bloki programowe (PB)**
  wykorzystywane do strukturyzacji programu użytkownika,
- **bloki funkcyjne (FB)**
  pozwalają na wykorzystanie całego zbioru dostępnych instrukcji procesora i są wywoływane razem z argumentami. Umożliwia to programowanie wielokrotnie powtarzających się sekwencji z różnymi wartościami parametrów,
- **bloki sekwencyjne (SB)**
  są szczególną formą bloków programowych. Pozwalają na programowanie zadań sterowania sekwencyjnego – odpowiadają następującym po sobie zadaniom sterowanego procesu; sekwencja taka składa się z bloków-zadań i warunków przejścia (tranzycji),
- **bloki danych (DB)**
  są w nich przechowywane stałe i zmienne dane wykorzystywane w programie, np. zmierzona przez sensor wartość ciśnienia (**rys. 2**).

Podobnie jak w innych językach programowania należy zwracać uwagę, aby uzależniać między sobą tylko zmienne jednakowego typu. Typ danych określa zarówno zbiór wartości jakie mogą one przyjmować, jak i zbiór operacji, który może być nad nimi wykonany. Na przykład typ BOOL określa daną w postaci jednego bitu – może to być zmienna wejściowa I 5.2, zmienna wyjściowa Q 0.2 lub marker M 100.0. Zmienne o typie danych BYTE, WORD lub DWORD są ciągami bitów, odpowiednio zmienną zawierającą 8 kolejnych bitów, liczbą całkowitą zajmującą 16 bitów o zakresie wartości od –32768 do +32767 oraz liczbą całkowita podwójnej precyzji zajmującą 32 bity o zakresie wartości od –2147483648 do +2147483647. Przegląd częściej wykorzystywanych typów danych zebrany jest w **tabl. 1**.



Rys. 1. Strukturyzacja programu



Rys. 2. Blok programowy – przykład

Tabl. 1. Elementarne typy danych

typ danych	opis		przykład
BOOL	bit	1 Bit	TRUE FALSE
BYTE	bajt (byte); liczba 8-bitowa w kodzie szesnastkowym	8 Bitów	B#16#00 B#16#FF
WORD	słowo liczba 16-bitowa w kodzie szesnastkowym wartość licznika 3 cyfry w kodzie BCD	16 Bitów	W#16#0000 W#16#FFFF 2#0000_0000_0000_0000 2#1111_1111_1111_1111 C#000 C#999
INT	liczba stałoprzecinkowa	16 Bitów	-32768 +32767
DINT	liczba stałoprzecinkowa	32 Bity	L#-2 147 483 648 L#+2 147 483 647
REAL	liczba zmiennoprzecinkowa	32 Bity	-123.4567 +1.234567E+02
TIME	wartość czasu w formacie IEC	32 Bity	TIME#24d20h31m23s647ms TIME#24d20h31m23s647ms
S5TIME	wartość czasu	16 Bitów	S5T#3ms S5TIME#2h46m30s

Powtórzenie i utrwalenie

1. **Proszę opisać sposób działania sterownika PLC.**
2. **Jakie – co najmniej – moduły zawiera sterownik PLC o budowie modułowej?**
3. **Co rozumiemy pod pojęciem stanu procesu?**
4. **Jakimi obszarami pamięci dysponuje sterownik PLC? Jakie operandy są przechowywane w jego pamięci systemowej?**
5. **Proszę podać grupy języków programowania PLC zalecane przez normę IEC 1131.**
6. **Co rozumie się pod pojęciem adresowania symbolicznego?**
7. **Przy pomocy jakich elementów strukturyzuje się program użytkownika PLC.**

Źródło: fragment książki „Mechatronika” pod kierunkiem D. Schmida, wyd. REA 2002 r.